

Towards Network-Failure-Tolerant Content Delivery for Web Content

Wen Hu¹, Zhi Wang², and Lifeng Sun¹

¹Tsinghua National Laboratory for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University

²Graduate School at Shenzhen, Tsinghua University
{hu-w12@mails., wangzhi@sz., sunlf@}tsinghua.edu.cn

Abstract—Popularly used to distribute a variety of multimedia content items in today’s Internet, HTTP-based web content delivery still suffers from various content delivery failures, including server failures [1], network failures [2] and routing failures [3]. Hindered by the expensive deployment cost, the conventional CDN can not deploy as many edge servers as possible to successfully deliver content items to all users under these delivery failures. In this paper, we propose a joint CDN and peer-assisted web content delivery framework to address the delivery failure problem. Different from conventional peer-assisted approaches for web content delivery, which mainly focus on alleviating the CDN servers’ bandwidth load, we study how to use a browser-based peer-assisted scheme, namely WebRTC, to resolve content delivery failures. To this end, we carry out large-scale measurement studies on how users access and view webpages. Our measurement results demonstrate the challenges (e.g., peers stay on a webpage extremely short) that can not be directly solved by conventional P2P strategies, and some important webpage viewing patterns (e.g., predictability of users’ webpage viewing time). Due to these unique characteristics, WebRTC peers open up new possibilities for helping the web content delivery, coming with the problem of how to utilize the dynamic resources efficiently. We formulate the peer selection that is the critical strategy in our framework, as an optimization problem, and design a heuristic algorithm based on the measurement insights to solve it. Our simulation experiments driven by the traces from Tencent QZone, one of the most popular social service platforms in China, demonstrate the effectiveness of our design: compared with non-peer-assisted strategy and random peer selection strategy, our design significantly improves the successful relay ratio of web content items under network failures, e.g., our design improves the content download ratio up to 60% even when users located in a particular region (e.g., city) where none can connect to the regional CDN server.

I. INTRODUCTION

Dominating today’s Internet traffic, HTTP-based online services are suffering from many types of content delivery failures (e.g., server failures [1], network failures [2], routing failures [3]), significantly degrading the quality-of-service (QoS) provisioned to online users. For example, Google service’s outage led to a traffic drop of around 40% during 5 minutes in 2013. This problem has been exacerbated when today’s online services are designed to utilize aggregated content items and information, which are served by a geo-distributed Content Delivery Network (CDN), e.g., a web page consisting of content items that are served by multiple servers may fail to

render at the client side if it fails to download only parts of the content items from some servers.

Conventional approaches to solve this problem are summarized as follows: (1) Replicating content items to multiple servers at different locations to avoid the failures of the servers [4], e.g., a user will be directed to download the same content from a different server when the original requested server does not respond. (2) Deploying edge-network proxies [5], e.g., a user can download content items from the nearby gateway proxy which may cache the requested content items. However, such approaches: (1) require additional replication of content items, causing additional deployment and operation costs; (2) are primarily for resolving the problem of static content items delivery; (3) are infeasible for today’s websites, which contain not only static content items, such as images, videos and SWF (Flash objects), but also dynamic content items [6] which include personalized elements—intrinsic un-cacheable feature of dynamic content items degrades the performance of these approaches.

In this paper, we propose a joint CDN and peer-assisted web content delivery framework based on WebRTC [7], which is defined by World Wide Web Consortium (W3C) and the Internet Engineering Task Force (IETF), and has been widely supported by major browser vendors [8], e.g., Google, Opera, Mozilla, Microsoft (under development). As illustrated in Fig. 1, by strategically selecting appropriate WebRTC-powered browsers to form the “recovery” delivery path (denoted as green dotted lines), these widely deployed browsers can assist in content delivery when a user fails to receive requested content items from the original CDN servers. As opposed to the conventional content delivery path, the recovered delivery path is not only from the CDN server through backbone networks to end users, but also from end users to end users. In order to select the most appropriate peer to recover content delivery failures, we perform exhaustive measurement studies on users’ webpage viewing behaviors, and the web-session characteristics.

Our contributions in this paper are summarized as follows:

- ▷ We conduct large-scale measurement studies to demonstrate the content delivery failures in today’s web content delivery to motivate our design. We also study users’ webpage viewing behaviors and the web-session characteristics,

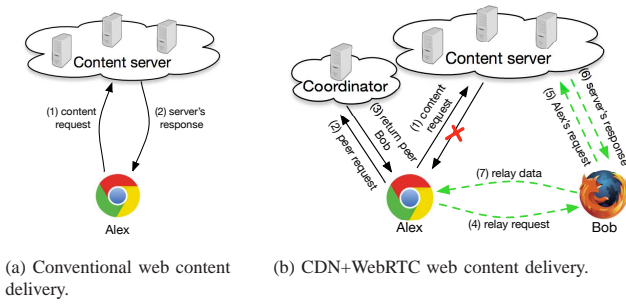


Fig. 1: Comparison between conventional content delivery and CDN+WebRTC content delivery.

including: (1) high churning rate of online peers; (2) low probability of “re-join”, i.e., re-visit the same webpage later; (3) fragmentation of web content, i.e., various content types (e.g., JavaScript, HTML, CSS, image, Flash, etc) are placed on different servers. These motivate us to propose a novel peer selection strategy customized for our CDN and peer-assisted web content delivery design, but not directly adopt the traditional P2P strategies.

▷ Based on our measurement studies, we propose a joint CDN and peer-assisted web content delivery framework and mathematically formulate the peer selection problem as an optimization problem. The objective is to choose a set of peering WebRTC users to form a delivery path in case of network failures. A heuristic algorithm is designed to solve this problem.

▷ We carry out simulation-based experiments to test the effectiveness of our design under dynamic and extreme running scenarios. Our design improves the content download ratio up to 60% even when users located in a region (e.g., a city) where none can connect to the regional CDN server.

The rest of the paper is structured as follows. In Section II, we carry out some measurement studies to demonstrate the motivation and design principles. We present the problem formulation and our design in Section III. In Section IV, we evaluate the performance of our system. Section V discusses the related work. We conclude this work in Section VI.

II. MEASUREMENT-DRIVEN MOTIVATION AND DESIGN PRINCIPLES

In this section, we present our motivation, and the design principles learnt from measurement studies.

A. Measurement Methodology

We first present how we carry out the measurement studies. We study users’ webpage accessing patterns based on real world traces collected by country-wide deployed CDN servers of Tencent QZone [9]. The traces consist of the user visiting information of two testing webpages in Tencent QZone, including 118,707 webpage visits, in which 2,300 sessions have encountered network failures. Each trace item contains the following information: (1) the user identifier, (2) the timestamp when a user begins to request the webpage, (3) the timestamp when a user leaves the current webpage, (4) the indicator of content fetching failure (i.e., whether a content item contained

in a webpage can not be downloaded from the original server). Based on these traces, we are able to study the challenges in peer-assisted web content delivery and the design principles.

B. Strong Randomness of Web Content Delivery Failures

In our collected traces, we analyze the failure distribution in terms of the time and the locations. In Fig. 2(a), we plot the number of content download failures recorded by our traces in each timeslot (6 hours). We observe that download failures may happen at different time of a day. Next, we plot the cumulative number of regions (city level) where users suffer from download failures over time in Fig. 2(b). We observe that the number of cumulative locations where users have experienced network failures keeps increasing, indicating that the failure locations are geographically distributed, i.e., users located at many places may suffer from download issues.

These observations indicate that the randomness of the content delivery failures makes the traditional replication/caching strategies based content delivery schemes no longer applicable.

Next, we study the design principles for our joint CDN and WebRTC-assisted webpage delivery framework, by measuring users’ webpage accessing patterns.

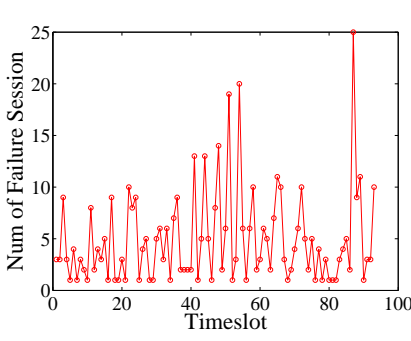
C. Webpage Viewing Patterns

In our design, the peer delivery assistance is based on the implementation of WebRTC, which requires users to stay on a webpage so as to peer in content delivery. Thus, users’ webpage viewing patterns have a significant impact on peers’ resource availability.

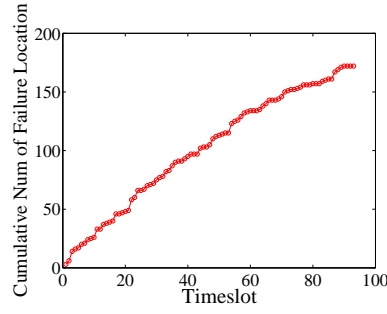
1) *Extremely Short Session Duration*: As recorded in our collected traces, each webpage viewing event is defined as a “session”. We first study the duration of the webpage viewing session, i.e., the time users spend on staying at a webpage. We have sampled 97,905 users viewing two webpages (referred as Webpage A and Webpage B) of different types. In Fig. 3, we plot the CDF of session durations of users viewing these two webpages on August, 2014.

We have made the following observations: Compared with traditional P2P services and applications, webpage sessions have much shorter durations, i.e., over 60% (resp. 90%) of session durations are shorter than 1 minute (10 minutes). In a P2P content sharing service, how long peers stay in the system determines the level of peer resource [10], since users are able to find more peering neighbors when peers stay longer. However, according to our observations of the extremely short webpage session durations, it is challenging for conventional P2P strategies to find enough peering resource to assist the web content delivery. Our design takes this short session characteristics into consideration and actively makes use of peers for content delivery assistance, by predicting when users will leave the webpage right after they open a webpage.

2) *Small Re-join Possibility*: In traditional P2P systems, peers re-joining the service may bring their cached data and serve other users. We also study the possibility that users re-join a webpage. As illustrated in Fig. 4, the curve is the CDF of the number of visits from the same user to the same



(a) The number of failed sessions.



(b) The cumulative number of failed users' locations.

Fig. 2: Failure distribution over time. Each timeslot is six hours.

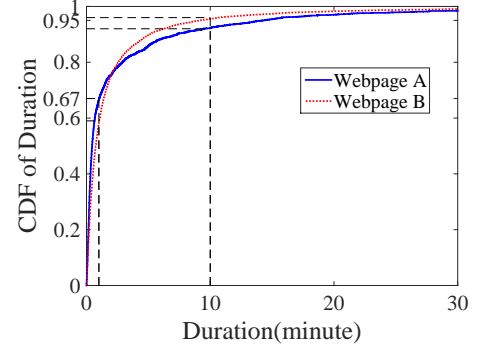


Fig. 3: Distribution of webpage sessions' duration.

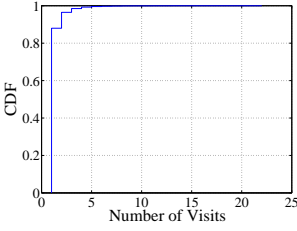


Fig. 4: CDF of number of visits to the same webpage.

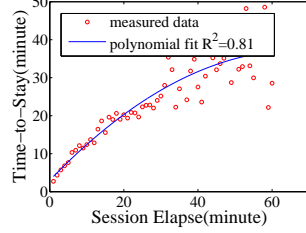


Fig. 5: *Time-to-stay* estimation for sessions *elapse* in $[0, 60]$.

webpage in 14 days. We observe that users hardly re-visit the webpages they have already viewed, e.g., less than 12% of the users will return to the same webpage in 14 days. The reason is that the two webpages in our experiments contain “static” content items, which do not change during our measurement period, and users do not re-visit these webpages to check out updates. As for webpages with more frequent updates, the re-join possibility may increase.

This observation indicates that compared to traditional P2P sharing systems, whose clients may return to systems up to tens of times per day (e.g., 60 times/day [11]), the possibility of webpage re-visits is much smaller. This also challenges our CDN and peer-assisted web content delivery design, especially for webpages updated less frequently.

3) *High Correlation Between Time-to-Stay and Session Elapse*: As the short session durations may have a negative impact on the performance of our proposed CDN and peer-assisted web content delivery framework, we next study how to tell which peers may stay long in the system and which peers may leave the webpage soon. We take an in-depth look at the webpage session durations in our dataset. By randomly splitting session durations into two parts: the *elapse* part, which is defined as the duration a user has already stayed on a webpage, and the *time-to-stay* part, which is defined as the duration the user will keep staying on the webpage, we generate a set of (*elapse*, *time-to-stay*) pairs of each session. We plot the *time-to-stay* versus the *elapses* of 330 samples from 30,000 pairs (i.e., the session *elapse* of 30,000 pairs fall into 330 *elapse-time* bins of one-minute length). Due to the space limitation, we do not show this figure. We observe

that for the short sessions, the *time-to-stay* is highly correlated with the *elapse* of the sessions, indicating that a user tends to keep staying on a webpage if he/she has already spent longer time on the webpage. Furthermore, we dive into more details for the short samples (i.e., duration < 60 minutes), which accounts for 97% of all the traces. As illustrated in Fig. 5, the polynomial model $y = -0.0076x^2 + 0.97x + 3.5$ fits well with these samples, further verifying the correlation between *time-to-stay* and *elapse*.

These observations and analysis indicate that we can accurately predict the *time-to-stay* of a webpage session, and select “stable” peers in our joint CDN and peer-assisted delivery design.

III. PATH-AWARE PEER-ASSISTED WEB CONTENT DELIVERY

In this section, we will analyze the peer selection problem in our CDN and peer-assisted framework. We formulate it as an integer programming problem, and design a heuristic algorithm to solve it.

A. Framework

We propose a path-aware peer-assisted web content delivery framework, where WebRTC peers are utilized to help other users fetch web content items from the web servers, to maximize the successful delivery ratio and the system throughput. We compare our content delivery framework with the conventional web content delivery paradigm in Fig. 1. Fig. 1(a) presents the conventional content delivery paradigm, where users directly download content items contained in a webpage from the corresponding CDN servers. Fig. 1(b) presents our peer-assisted delivery paradigm: In the case of delivery failures (denoted as a red cross), a user (e.g., Alex) will try to receive web content items from other WebRTC peers (e.g., Bob) by a recovered path (denoted as green dotted lines).

B. Problem Formulation

The main idea behind our design is to strategically assign the WebRTC peers, which have available peering resource (e.g., uplink bandwidth capacity), to users who encounter

delivery failures. As such, generating a candidate peer list through carefully peer selection for each failed user is critical in our design. Before diving into more details, we present several definitions used in our formulation.

1) *P2P Connectivity and Bandwidth Matrix*: First, we define a peer-to-peer bandwidth matrix $\mathbf{B}^{M \times N}$ and each entry $b_{ij}(t)$ represents the estimated upload capacity from a WebRTC peer j to user i at time t .

2) *Peer Selection Strategy*: Next, we define our peer selective strategy as a peer selection matrix $\mathbf{P}^{N \times M}$ and each entry is a binary variable, i.e., $p_{ij}(t) = 1$ indicates we choose peer i as the relay peer for peer j at time t ; otherwise, $p_{ij}(t) = 0$. In our design, the peer selection is generated over time, i.e., the peer selection matrix \mathbf{P} changes over time according to the estimated P2P bandwidth. We assume that a user only utilizes one relay peer when downloading a single content. However, our design can also be extended to use multiple relay peers to fetch a web content.

3) *Optimization Formulation*: We formulate the peer selection problem as an optimization problem, as follows.

$$\max_{p_{ij}(t)} \sum_{i=1}^n \sum_{j=1}^m p_{ij}(t) b_{ji}(t) \quad (1)$$

subject to,

$$\sum_{i=1}^n p_{ij}(t) \geq 1 \quad \forall j \in \{1, \dots, m\} \quad (2)$$

$$\sum_{j=1}^m p_{ij}(t) b_{ji}(t) \leq bw_i(t) \quad \forall i \in \{1, \dots, n\} \quad (3)$$

The rationale of the optimization is to find a “match” between relay peers and the users encountering delivery issues, to maximize the successful relay ratio and the throughput of delivering the failed content items. Constraint (2) guarantees that there is at least one peer available for the requesting user. Constraint (3) guarantees that the relay peer can not serve many requests exceeding its own uplink capacity.

C. Path-Aware Peer Selection Strategy

Our heuristic and distributed algorithm works as follows: (1) Based on various peer selection factors, we generate a peer list for a requesting user; (2) The requesting user then actively tries these candidate peers to download the failed web content. We next elaborate these two steps, respectively.

1) *Factors for Peer Selection*: First, since the real-time measurement of network state is too expensive and the geographic location and the ISP of the peer provide valuable implications for the network state [12], in our design, we select relay peers that are within the same location and Internet Service Providers (ISP) with the requesting peers in order to achieve a better network performance. In particular, the same ISP also reduce the cross ISP cost [13].

Moreover, due to the extremely short duration of web viewing sessions, it is of paramount importance to guarantee that the relay peer will still be online when it is chosen to

serve the requesting peer. Thus, we have to estimate how long a peer will be online. In our measurement study in Section II-C3, we observe the fact that the *time-to-stay* of a session is highly correlated with the *elapse* of the session. Based on this observation, we prioritize candidate relay peers according to the duration they have already been viewing a webpage. As such, the selected relay peers are more likely to be online during the content delivery phase.

Besides, according to our measurement studies, content delivery failures show randomness with respect to the time and the locations. In order to increase the diversity of potential content delivery paths, we (1) select relay peers who are at different locations to overcome the regional network failure; (2) choose relay peers with diverse ISPs such that relay peers may have an improved connectivity with the original content server; (3) select the peers with various load to achieve load balance among all relay peers in the system.

2) *Relay Peer List Generation*: Based on the factors discussed above, when there is a relay request from peer p_s , two steps are involved to generate the relay peer list.

- The requesting peer p_s chooses relay peer p_r from the online peer set, which is obtained from the coordinate server;
- Rank the list based on the workload and the *time-to-stay* of the candidate relay peers.

The peer selection algorithm carried out by the coordinate server is summarized in Algorithm 1. Given the input parameter ζ , which is the length of relay peer list, we generate the list consisting of two peer sets: the peers selected carefully (R_I) and the peers selected randomly (R_{II}). α is the ratio of each set and it is adjustable based on the specific network environment. Note that R_I consists of peers located in the same city and served by the same ISP with the requesting peer (line 3), to achieve a better network performance, and R_{II} consists of randomly selected peers (line 7), to increase the diversity of potential content delivery paths. We filter those relay peers which experienced some content fetching failures or a higher workload (i.e., how many peers it has been already relaying for) exceeding the threshold γ in the two sets (R_I, R_{II}) (line 10, line 11). Furthermore, in order to select the peers not only have more spared bandwidth and will stay online longer, we sort the two parts in the descending order of the *time-to-stay* τ_i (line 12).

3) *Content Relaying with Relay Peers*: The first peer in the candidate relay list acts as the *primary* peer, and the remaining peers act as backups. After obtaining the relay peer list, the content relaying works as follows: The requesting peer p_s first asks the primary peer to fetch the failed content, then it tries other candidates in the candidate list until the content is finally fetched.

IV. PERFORMANCE EVALUATION

We verify the effectiveness and performance of our design with simulation experiments driven by the traces presented in Section II.

Algorithm 1 Relay Peer List Generation.

```
1: procedure RELAY-PEER-SELECTION( $\alpha, \gamma, \zeta$ )
2:   for  $i = 0$  to  $\zeta * \alpha$  do
3:     select  $p_r$  which has the same location and ISP with
        $p_s$ , from online peer set
4:      $R_I \leftarrow R_I \cup p_r$ 
5:   end for
6:   for  $i = (\zeta * \alpha + 1)$  to  $\zeta$  do
7:     select  $p_r$  from online peer set randomly
8:      $R_{II} \leftarrow R_{II} \cup p_r$ 
9:   end for
10:  filter peers which experienced some content fetching
    failure in  $R_I, R_{II}$ , respectively
11:  filter peers whose workload exceed the threshold  $\gamma$  in
     $R_I, R_{II}$ , respectively
12:  sort  $R_I, R_{II}$  in descending order of  $\tau_i$ 
13:   $R \leftarrow R_I \cup R_{II}$ 
14:  return  $R$ 
15: end procedure
```

A. Experiment Setup

To study the details in its performance under extreme scenarios, we carry out simulation experiments with different settings. In our experiments, we simulate 5,000 peers that are distributed in five cities in China. Based on the geographic of the cities, i.e., the longitude and latitude information, we calculate the physical distance between each peer pair. The latency between peer pairs is estimated using the correlation between distance and latency [14]. To simulate the real Internet environment, the peers are configured heterogeneously in terms of uplink and downlink capacities, following the statistics in [15]. The ISP of each user is randomly assigned to 3 ISPs. According to the survey in [16], the average web page is 1,600 KB consisting of 112 objects. Here, we analyze the content size ranging from 500 KB to 16,000 KB due to the rapid growth of the web page size.

User behaviors: We simulate the arrival patterns as a poisson process with the $\lambda = 30$, and the session duration with the Pareto distribution observed in Section II. Relaying peers who are potential to help the failed peer fetch content are selected according to Algorithm 1. If not otherwise specified, $\alpha = 0.2, \gamma = 0.8, \zeta = 10$ [17].

Content delivery failures: In our experiments, we assume that users download web content items from regional CDN servers, in which users are redirected to a regional peering server to download the content items according to their locations and ISPs. We simulate in-network failures as follows. An in-network failure [18] is the case that the network of a user's region is encountering some issues, and a fraction of users in this region can not connect to the server. An example of such failure is illustrated in Fig. 7: user u_1 and user u_2 can not download content from the server and connect to each other; however, the other users/peers (e.g., u_3, u_4), are able to connect to both u_1, u_2 and the server. In our experiments, the in-network failure ratio is 60% if not otherwise specified.

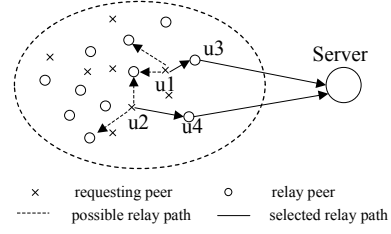


Fig. 7: Illustration of in-network failures.

Baselines: We compare our design with two schemes, i.e., (1) no relay: users directly download content items from CDN servers; (2) random peer selection: users request randomly selected peers to download failed content items; (3) our design: users download failed content items from the peers selected with our strategies. Next, we present the experiment results.

B. Performance under In-Network Failures

First, we study the successful relay ratio, which is defined as the ratio of requests that can be finally served by CDN servers or peers. As illustrated in Fig. 6(a), the curves represent the successful relay ratio of different strategies versus the web content size. We observe that our design outperforms the random peer selection strategy, and the no-relay strategy. In particular, when the content size is about 16 MB, our design improves the successful content download ratio by over 25% against the random relay selection approach.

Next, we investigate the impact of the in-network failure ratio on the successful relay ratio with the primary peer, i.e., the first relay peer in the candidate relay peer list. As illustrate in Fig. 6(b), the curves represent the successful relay ratio versus the in-network failure ratio. We observe that with the increase of in-network failure ratio, the successful relay ratio decreases. In particular, compared with our strategy, the successful relay ratio of random and no-relay strategy decrease much faster. When the in-network failure ratio reaches 100%, i.e., all users in a particular region can not connect to the CDN server, our design still achieves a final successful download ratio of 60%.

We further investigate the average repeated request number in the successful relay session, which is defined as the number of repeated requests from requesting users to the candidate relay peers until it successfully downloads the content. In Fig. 6(c), the bars represent the number of requests tried by requesting users versus the file size of the web content items. We observe that compared with the random peer selection, users try a much smaller number of relay requests to successfully download a failed content, and the performance gap is more remarkable when the file size increases. In particular, when the file size is 16 MB, it only takes about 2 tries for a user to download a content from the relay peers selected with our strategy, whereas it takes more than 7 tries with the random peer selection approach.

V. RELATED WORK

In this section, we survey related works on WebRTC-powered content delivery.

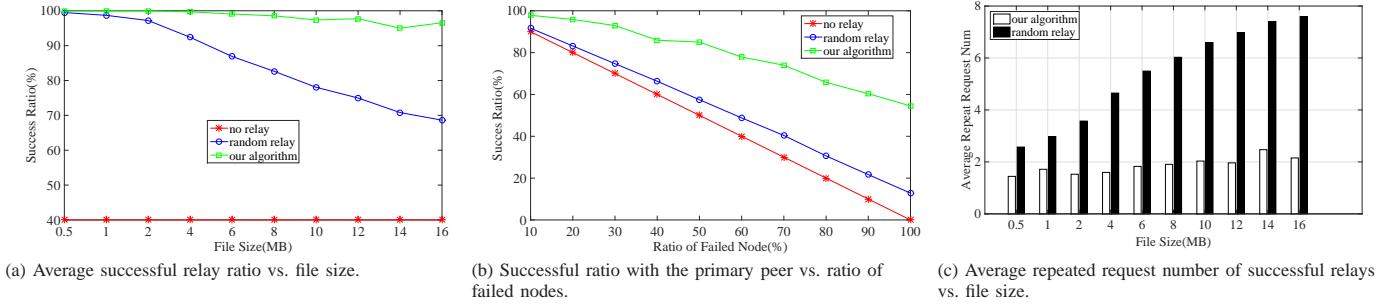


Fig. 6: The performance comparison among different algorithms when in-network failures occur.

WebRTC allows direct browser-to-browser communication, leading to growing peer-assisted web applications. To date, there are over 1 billion endpoints in use supporting WebRTC, and it is anticipated that 4.7 billion devices will support WebRTC by 2018 [8]. Some systems have been developed to utilize WebRTC browsers to assist in content delivery. Zhou et al. [19] developed WebCloud, which decentralized web content exchanging by utilizing users' browsers to deliver content; however, it required that redirector proxies had been deployed within each ISP region. Zhang et al. [20] proposed Maygh, a system that automatically recruited web visitors to help content servers, to reduce the cost for operating a web site.

The limitation of the previous works on WebRTC is that they mainly focused on alleviating the original content servers based on the P2P philosophy, in which peers share bandwidth with each other. However, our work focuses on seeking a solution to address the network failure problem [21].

VI. CONCLUSION

In this paper, we propose a joint CDN and peer-assisted web content delivery framework to address the content delivery failure problem for today's WebRTC-powered peers. Different from traditional peer-assisted approaches that mainly focus on alleviating the CDN servers' bandwidth load, our contribution in this paper lies in that we first study to use a browser-based peer-assisted scheme to resolve content delivery failure problem. Based on large-scale measurement studies on how users access and view webpages, we not only show the challenges in our design that can not be directly solved by conventional P2P strategies (e.g., peers stay on a webpage extremely short), but also learn webpage viewing patterns and design principles. In particular, we formulate the peer selection problem as an optimization problem, and design a heuristic algorithm based on the measurement insights to solve it. Our simulation experiments demonstrate the effectiveness of our design: compared with non-peer-assisted strategy and random peer selection strategy, our design significantly improves the successful delivery ratio under network failures.

REFERENCES

[1] "facebook engineering more details on todays outage," Facebook outage, 2010.

[2] D. Turner, K. Levchenko, A. C. Snoeren, and S. Savage, "California fault lines: understanding the causes and impact of network failures," *ACM SIGCOMM CCR*, 2011.

[3] D. Watson, F. Jahanian, and C. Labovitz, "Experiences with monitoring ospf on a regional service provider network," in *IEEE ICDCS*, 2003.

[4] S. Sivasubramanian, M. Szymaniak, G. Pierre, and M. v. Steen, "Replication for web hosting systems," *ACM Computing Surveys (CSUR)*, 2004.

[5] S. Ihm, *Understanding and improving modern web traffic caching*. Ph.D. Thesis, Princeton University, 2011.

[6] Z. Wang, W. Zhu, M. Chen, L. Sun, and S. Yang, "CPCDN: Content Delivery Powered by Context and User Intelligence," *IEEE TMM*, 2015.

[7] <http://www.w3.org/2011/04/webrtc-charter.html>, W3C Real-Time Communications Working Group.

[8] C. Alexandru, "Impact of webrtc (p2p in the browser)," *Internet Economics VIII*, 2014.

[9] "Tencent qzone," <http://qzone.qq.com/>.

[10] F. Wang, J. Liu, and Y. Xiong, "Stable peers: Existence, importance, and application in peer-to-peer live video streaming," in *IEEE INFOCOM*, 2008.

[11] D. Stutzbach and R. Rejaie, "Understanding churn in peer-to-peer networks," in *ACM IMC*, 2006.

[12] W. Hu, Z. Wang, and L. Sun, "Guyot: a hybrid learning-and model-based rtt predictive approach," in *IEEE ICC*, 2015.

[13] R. Bindal, P. Cao, W. Chan, J. Medved, G. Suwala, T. Bates, and A. Zhang, "Improving traffic locality in bittorrent via biased neighbor selection," in *IEEE ICDCS*, 2006.

[14] E. Katz-Bassett, J. P. John, A. Krishnamurthy, D. Wetherall, T. Anderson, and Y. Chawathe, "Towards ip geolocation using delay and topology measurements," in *ACM IMC*, 2006.

[15] M. Zhang, Y. Xiong, Q. Zhang, L. Sun, and S. Yang, "Optimizing the throughput of data-driven peer-to-peer streaming," *IEEE TPDS*, 2009.

[16] <http://www.websiteoptimization.com/speed/tweak/average-web-page>.

[17] N. Magharei and R. Rejaie, "Understanding mesh-based peer-to-peer streaming," in *ACM NOSSDAV*, 2006.

[18] M. Dahlin, B. Chandra, L. Gao, and A. Nayate, "End-to-end wan service availability," *IEEE/ACM ToN*, 2003.

[19] F. Zhou, L. Zhang, E. Franco, A. Mislove, R. Revis, and R. Sundaram, "Webcloud: Recruiting social network users to assist in content distribution," in *IEEE NCA*, 2012.

[20] L. Zhang, F. Zhou, A. Mislove, and R. Sundaram, "Maygh: Building a cdn from client web browsers," in *ACM EuroSys*, 2013.

[21] W. Hu, Z. Wang, and L. Sun, "Path-aware peer-assisted web content delivery against network failures," in *IEEE/ACM IWQoS*, 2015.